```
      program yaw_driver
c program to exercise gfo_yaw subroutine for Gipsy-Oasis II
      implicit double precision (a-h,o-z)
      dimension dpsi(2)
      open(unit=3,file='yaw.dat',status='new')
c loop over beta angles
      do 10 i=0,14
      beta=-90.d0+float(i)*20.d0
      if(i.eq.10) beta=-2.
      if(i.eq.11) beta=0.
      if(i.eq.12) beta=2.
      if(i.eq.13) beta=89.
      if(i.eq.14) beta=-89.
c     beta=-50.
c     write(3,4)beta
    4 format(//,' beta angle = ',f10.2,/)
c loop over orbit angles
      do 10 j=0,180
      omega=float(j)*2.d0
      call gfo_yaw(beta,omega,psi,dpsi)
      if(j.gt.0) then
        if(psi-psi_last.gt.90.) psi=psi-360.
        if(psi-psi_last.lt.-90.) psi=psi+360.
      endif
      psi_last=psi
      write(3,8)beta,omega,psi,dpsi
    8 format(3f10.2,2f10.5)
   10 continue
      stop
      end



      subroutine gfo_yaw(beta,omega,psi,dpsi)
c
c** Subroutine to compute GFO yaw angle for use by Gipsy/Oasis II
c  orbit determination software.  Input and output are the same
c  as topex_yaw, which this subroutime will replace for GFO.
c
c  Written by Scott Mitchell, December, 1995.
c  Yaw algorithm from Doug Wiemer.
c
      implicit none
      character*60 SccsID
c
c** Input variables
      double precision beta
c The inclination of the earth-sun vector wrt the orbit plane, in degrees.
c It is positive if the sun and the north pole are on one side of the
c orbit plane, and negative otherwise.
      double precision omega
c The orbit angle, in degrees, measured counter-clockwise from a point
c on the orbit plane 90 degrees clockwise from the sun direction.  When
c the s/c is closest to the sun, omega is pi/2.  omega is always in the
```

```
c range [-180,180].
c
c** Output variables
      double precision psi      ! The yaw angle, in degrees.
      double precision dpsi(2)  ! Partials of psi wrt beta and omega, respect.
c Note that yaw is the angle between the velocity direction (x axis in orbit
c frame) and the spacecraft x axis (the long axis of GFO), positive as a
c positive rotation about the z axis in the orbit frame (nadir).
c
c** Local variables
      integer i,j
      double precision x0(3)    ! x axis of the orbit frame (velocity dir)
      double precision z0(3)    ! z axis of orbit frame (nadir)
      double precision offset   ! yaw bias due to altimeter pointing
      double precision sun(3)    ! sun in orbit frame
      double precision sn(3)    ! normal to sun-nadir plane
      double precision snm      ! magnitude of sn
      double precision betar    ! beta in radians
      double precision omegar   ! omega in radians
      double precision dd       ! increment of beta and omega for derivatives
      double precision psi1     ! yaw used for derivatives
      double precision pi, deg2rad, rad2deg, beta1
      data SccsID /'@(#) gfo_yaw.f   1.0     12/07/95 \0 '/
      data (x0(i),i=1,3)/1.d0, 0.d0, 0.d0/
      data (z0(i),i=1,3)/0.d0, 0.d0, 1.d0/
      data offset/6.1928d0/
      data dd/0.01d0/
c
c
      pi=2.d0*asin(1.d0)
      deg2rad=pi/180.d0
      rad2deg=180.d0/pi
c
      do 20 j=1,3
c  If j=1, calculate yaw angle phi
c  If j=2, calculate dpsi(1)=yaw angle derivative wrt beta
c  If j=3, calculate dpsi(2)=yaw angle derivative wrt omega
c
c  Local copy of beta
      beta1=beta
c
c  Offset apparent sun position if sun is too close to orbit plane
      if(beta.ge.0.d0 .and. beta.lt.3.d0) then
        beta1=3.d0
      elseif(beta.lt.0.d0 .and. beta.gt.-3.d0) then
        beta1=-3.d0
      endif
c
      betar=beta1*deg2rad
      omegar=omega*deg2rad
c
      if(j.eq.2) betar=betar+dd*deg2rad
      if(j.eq.3) omegar=omegar+dd*deg2rad
c
```

```
c  Calculate sun position in orbit frame
      sun(1)=cos(betar)*cos(omegar)
      sun(2)=sin(betar)
      sun(3)=cos(betar)*sin(omegar)
c
c  Normal to sun-nadir plane as cross product
      sn(1)=sun(2)*z0(3)-sun(3)*z0(2)
      sn(2)=-sun(1)*z0(3)+sun(3)*z0(1)
      sn(3)=sun(1)*z0(2)-sun(2)*z0(1)
c
c  Normalize
      snm=dsqrt(sn(1)*sn(1)+sn(2)*sn(2)+sn(3)*sn(3))
      do 10 i=1,3
   10 sn(i)=sn(i)/snm
c
c Yaw angle calculated as dot product
      psi1=acos(sn(1)*x0(1)+sn(2)*x0(2)+sn(3)*x0(3))*rad2deg*
     .   sn(2)/dabs(sn(2))
c
      if(j.eq.1) then
c Yaw angle calculation
       psi=psi1
      elseif(j.eq.2) then
c Yaw angle derivative wrt beta
       dpsi(1)=(psi1-psi)/dd
      elseif(j.eq.3) then
c Yaw angle derivative wrt omega
       dpsi(2)=(psi1-psi)/dd
      endif
c
   20 continue
c
c Add offset to yaw angle
      psi=psi-offset
c
      return
      end
```